necessarily subsume any other column of A_i since duplicate columns are removed in Step 3-b.3. However, at some point in the reduction of P' to I', P' is reduced to some $I'y_j^*$, $P' \subseteq I'y_j^*$ and, as shown above, there exists an FPI Q such that Q is the disjunction of $Q'\bar{y}_j^*$ and possibly X_i and/or $\bar{X}_i$. Q' may also have been reduced to Q'' by Step 3-b, $Q' \subseteq Q''$. So now,

$$A_i = X_i\bar{X}_i \begin{bmatrix} & \text{col 1} & \text{col 2} & \\ \cdots & I' & Q'' & \cdots \\ & y_j^* & \bar{y}_j^* & \end{bmatrix}$$

where col 1 is derived from P and col 2 is derived from Q. Since we are interested in generating $I'X_i\bar{X}_i$, we will consider two cases.

Case 1: $I' \subset Q''$. If P contains $X_i\bar{X}_i$, then an e is placed below col 1 and $X_i\bar{X}_iI' = I$ is added to A. However, $P = X_i\bar{X}_iP' \subseteq X_i\bar{X}_iy_j^*I' \subset X_i\bar{X}_iI' = I$. Therefore, $P \subset I$, which is a contradiction since P is an FPI. Therefore, P cannot contain $X_i\bar{X}_i$. So an $\tilde{e}$ is placed below col 1, and I is not added to A.

Case 2: $I' = Q''$. As shown above, P does not contain $X_i\bar{X}_i$. If Q contains $X_i\bar{X}_i$, then an e is placed below col 1 and col 2, and $X_i\bar{X}_iI' = I$ is added to A. However, $Q = X_i\bar{X}_i\bar{y}_j^*Q' \subseteq X_i\bar{X}_i\bar{y}_j^*Q'' \subset X_i\bar{X}_iQ'' = X_i\bar{X}_iI' = I$. Therefore, $Q \subset I$, which is a contradiction since Q is an FPI. Therefore, Q cannot contain $X_i\bar{X}_i$. So an $\tilde{e}$ is placed below col 1 and col 2, and I is not added to A.

Therefore, any nonessential FPI of F will not appear in the function resulting from the algorithm of this paper. Q.E.D.

V. Conclusions

In this paper we have described an algorithm which finds precisely the set of essential fuzzy prime implicants of a fuzzy function in a sum of products form. At present, this is the only known fuzzy minimization method for disjunctive functions which does so without the construction of a cover table.

This algorithm is less cumbersome than existing methods of generating all fuzzy prime implicants in that it does not include an intermediate expansion of the function. Reduction takes place only on those phrases which are in the original representation of the function. For this reason we feel that this is a significant algorithm, introducing concepts which may prove useful in other areas of fuzzy set theory.

References

[1] S. R. Das, "An approach for simplifying switching functions by utilizing cover table representation," *IEEE Trans. Computers* (Short Notes), vol. C-20, pp. 355–359, 1971.

[2] S. R. Das and N. S. Khabra, "Clause-column table approach for generating all the prime implicants of switching functions," *IEEE Trans. Computers*, vol. C-21, pp. 1239–1246, 1972.

[3] A. Kandel, "On minimization of fuzzy functions," *IEEE Trans. Computers*, vol. C-22, pp. 826–832, 1973.

[4] ——, "On the minimization of incompletely specified fuzzy functions," *Information and Control*, pp. 141–153, 1974.

[5] ——, "On the Properties of fuzzy switching functions," *Journal of Cybernetics*, pp. 119–126, 1974.

[6] ——, "A note on the simplification of fuzzy switching functions," Computer Science Report 139, New Mexico Institute of Mining and Technology, Socorro, New Mexico, 1975.

[7] ——, "Inexact switching logic," *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-6, pp. 215–219, 1976.

[8] A. Kandel and H. A. Davis, "The first fuzzy decade (Bibliography on fuzzy sets and their applications)," Computer Science Report 140, New Mexico Institute of Mining and Technology, Socorro, New Mexico, 1976.

[9] T. P. Neff and A. Kandel, "Simplification of fuzzy switching functions," to appear in the *International Journal of Computer and Information Sciences*, 1977.

[10] S. M. Rickman and A. Kandel, "Column table approach for the minimization of fuzzy functions," Computer Science Report 137, New Mexico Institute of Mining and Technology, Socorro, New Mexico, 1975.

[11] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353. 1965.

Two Modifications of CNN

IVAN TOMEK

***Abstract*—The condensed nearest-neighbor (CNN) method chooses samples randomly. This results in a) retention of unnecessary samples and b) occasional retention of internal rather than boundary samples. Two modifications of CNN are presented which remove these disadvantages by considering only points close to the boundary. Performance is illustrated by an example.**

Introduction

The condensed nearest-neighbor (CNN) method [1] is a method of preprocessing of the design set for pattern recognition. It is based on the nearest-neighbor (NN) rule [2]. Its purpose is to reduce the size of the original design set D (set of samples with known membership) by the elimination of certain samples without affecting significantly the performance of NN classification: The NN rule used with E (the new design set $E \subset D$), should give almost the same result as NN used with D.

CNN works as follows:

a) pass ← 1,
b) choose $x \in D$ randomly, $D(1) = D - \{x\}$, $E = \{x\}$,
c) D (pass + 1) = $\varnothing$, count ← 0,
d) choose $x \in D$ (pass) randomly, classify x by NN using E,
e) if classification found in d) agrees with actual membership of x
   then D(pass + 1) = D(pass + 1) $\cup \{x\}$
   else $E = E \cup \{x\}$, count ← count + 1,
f) D(pass) = D(pass) $- \{x\}$,
g) if D(pass) $\neq \varnothing$ go to d),
h) if count = 0
   then end of algorithm
   else pass ← pass + 1, go to b).

It is clear that CNN has the following properties. It generates a design set E 1) which is a subset of the original design set and 2) which classifies (NN rule) all samples in D correctly. Property 1) usually means that E is much smaller than D and thus computationally much better suited for NN classification: it requires less storage and computation. Property 2) indicates that NN classification with E is very similar (although not necessarily identical) to NN classification with D. This is especially true when D is "representative" (by this we mean that the number of samples and their distribution is such that the approximation of the "true" underlying probability distribution by relative frequency of samples is "good").

The disadvantage of CNN is that it processes samples from D randomly—moves them from D to E quite randomly at the beginning and less so later on (when it tends to take samples closer to the boundary). This means that E contains a) interior samples which could be eliminated completely without change in the performance and b) samples which define a boundary on E but not on D (i.e., samples not essential in D become boundary points in E). Point a) implies that E is larger than necessary, point b) causes an undesirable shift between boundaries.

The ideal method of reduction of D would work essentially as CNN but would only use points close to the decision boundary to generate E. Unfortunately, the true decision boundary is unknown by definition. The next best is to use only those points which generate the piecewise-linear decision boundary in D (as given by the application of the NN rule). Even this is difficult.

Manuscript received March 12, 1976; revised June 9, 1976.
The author is with the Department of Computer Science, Acadia University, Wolfville, NS, Canada B0P 1X0.

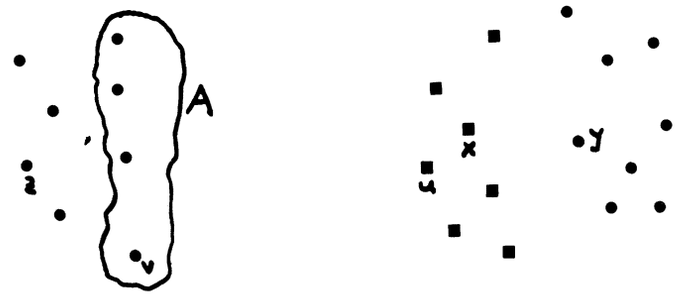

Fig. 1. Choice of boundary points—Method 1.

Fig. 2. Points represented by full symbols should be chosen for the final design set. They are not retained by Method 1.

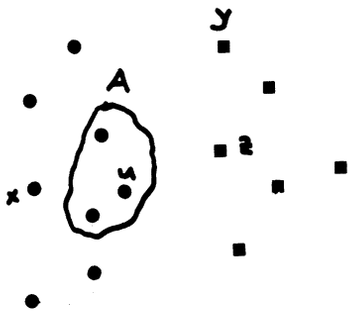

Fig. 3. Step e) in Method 1. y is the nearest neighbor of x ($y \in F$).

Two even less ideal methods will be described in the remainder of this text. It will be shown that they are considerable improvements upon CNN. They are both based on intuitive approximations of the notion of a boundary point.

Ordered CNN

Two methods will now be described which differ from CNN in that only samples with certain properties are considered for the reduced design set E.

Method 1

Let $x \in D$, y its nearest neighbor from the *opposite* class ($y = nno(x)$). Then y must be close to the decision boundary (Fig. 1). These points form the basis of our first modification of CNN. It has to be noted that they are not by themselves sufficient in all situations. If we only used them, points from set A (Fig. 1) would never be found, and yet they are necessary for E to classify all D correctly. To discover points in A we have to proceed indirectly. If at a given stage of generation of E, z is classified incorrectly (because no points from A are in E), use the nearest neighbor of z in E (u in Fig. 1). It belongs to the "opposite" class (since z is classified incorrectly and z is the nearest neighbor). Now find the nearest neighbor of u which classifies z correctly—this is $v \in A$.

Let us note that while this approach generates E which a) classifies D correctly and b) contains only boundary points, it still does not guarantee that all the desirable boundary points are included (Fig. 2). The algorithm is as follows:

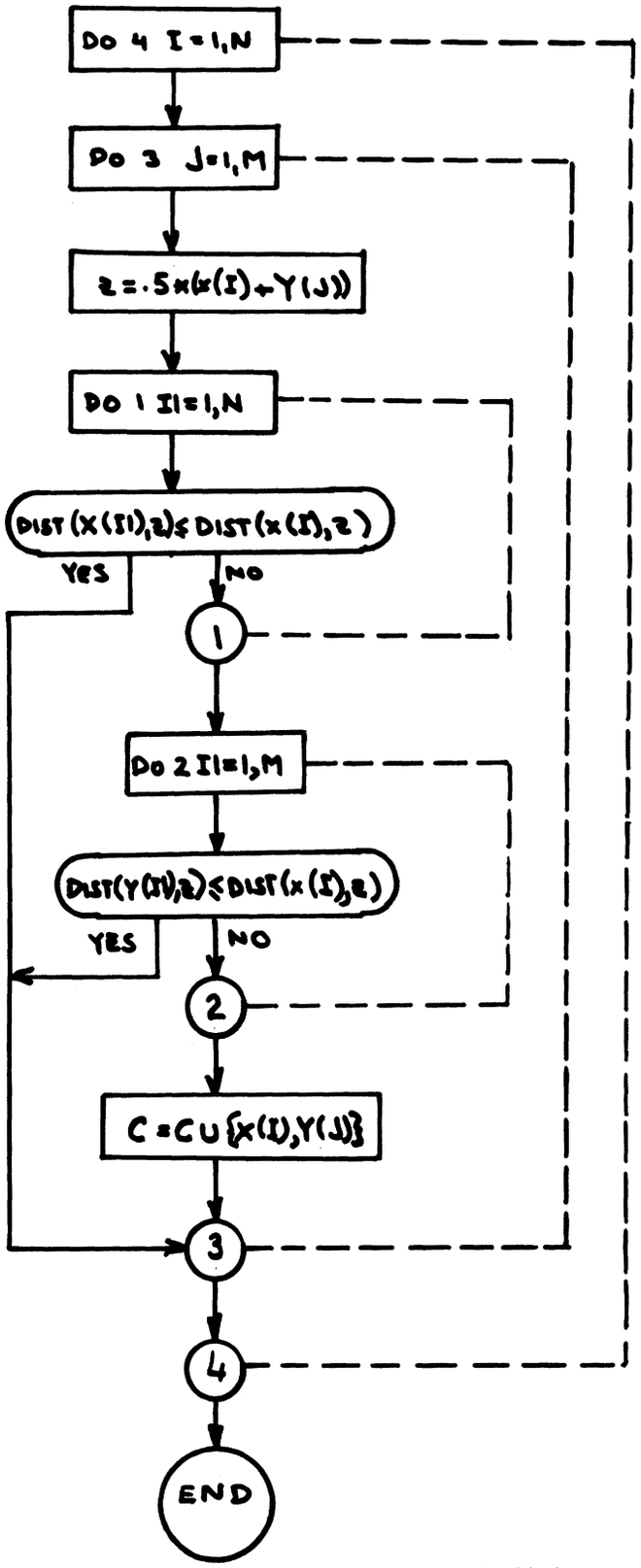

Fig. 4. Flowchart for Method 2 (notation adapted from [3]). C is the set of pairs of samples accepted for the final design set. $C = \varnothing$ at the beginning.

a) pass = 1,
b) choose $x \in D$ randomly, find $y = nno(x)$, $D(1) = D - \{y\}$, $E = \{y\}$, $F \neq \varnothing$,
c) $D(\text{pass} + 1) = \varnothing$, count = 0,
d) choose $x \in D(\text{pass})$ randomly and classify it by NN using E,

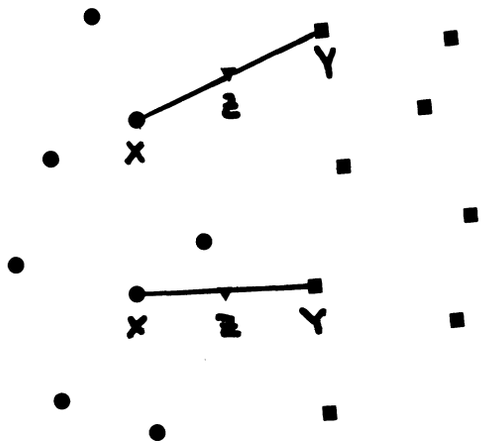

Fig. 5. Example of application of Method 2: The upper pair (X,Y) is accepted as a member of C, the lower pair is not.

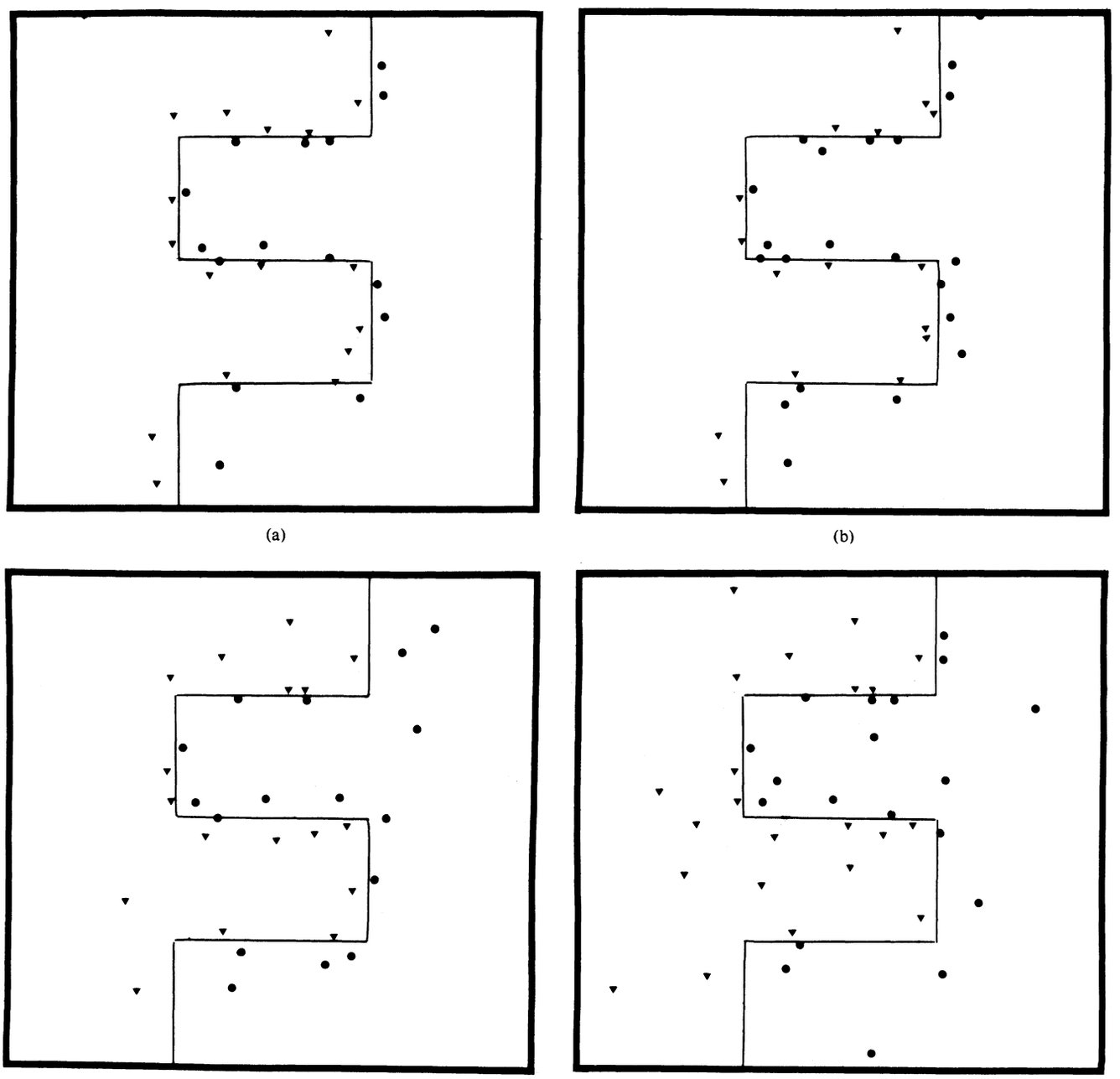

Fig. 6. Comparison of proposed methods with two others. Example taken from [4] with two classes with uniform distributions separated by the indicated boundary. Original set (400 samples) is not shown. Results of processing: (a) Method 1 (only subset C is shown). (b) Method 2. (c) Method from [4]. (d) CNN.

e) if classification found in d) agrees with actual membership of x,
then $D(\text{pass} + 1) = D(\text{pass} + 1) \cup \{x\}$
else if $x \in F$
then $E = E \cup \{x\}$, $F = F - \{x\}$
else classify x by F,
if classification agrees
then $E = E \cup \{x\}$, $F = F - \{x\}$
else find $z = nno(x)$, z D(pass) and assign it to F: $F = F \cup \{z\}$, next find u such that dist $(u,z) = \min v \in A$ dist (v,z) where $A = \{w \mid w \in D(\text{pass}), \text{dist}(x,w) < \text{dist}(x,z), \text{class}(w) = \text{class}(x)\}$, $E = E \cup \{u\}$.

Steps f), g), and h) are the same as in CNN.

A part of step e) is illustrated in Fig. 3. It is clear that Method 1 guarantees that E classifies all samples of D correctly. (Note that set A in step e) is always nonempty since $x \in A$.)

Method 2

The method works as CNN but instead of moving to E samples from the complete D, only a subset of D ($C \subset D$) is used. Subset C is chosen according to the flowchart in Fig. 4. (It is assumed that $x(i)$, $i = 1, N$ are all samples of D belonging to class 1 and $y(i)$, $i = 1, M$ all samples of D belonging to class 2.) See also Fig. 5.

Experimental Results

For illustration and comparison with other methods, an example presented in [4] has been repeated. Results are shown in Fig. 6. The original design set consists of 2 classes of 200 two-dimensional samples each. Distribution is uniform with irregular decision boundaries as indicated.

Conclusion

Both proposed modifications work better than CNN since a) the resulting design set is smaller and b) the retained boundary points are better chosen (close to the decision boundary). Method 2 has another potentially important property: It finds *pairs* of boundary points which participate in the formation of the (piecewise-linear) boundary. This information might be very useful in the development of more powerful methods of classification by piecewise-linear classifiers. Such methods could use these pairs to generate progressively simpler descriptions of acceptably accurate approximations of the original completely specified boundaries. Note also that the choice of subset C is order independent.

Unlike the other existing methods of reduction of the design set [4], [5], the proposed methods explicitly seek to find boundary points. This results in retaining points closer to the boundary and also in retaining fewer points.

Wilson [6] introduced an editing method intended to improve the classification. As a by-product, this method also insignificantly reduces the size of the design set (by eliminating most samples on the "wrong side" of the boundary). When, however, the edited set is processed by CNN or other methods, the result is reduction much more significant than that attainable on the original set. The reason for this is that the edited design set is much "cleaner" than the original one. This is also discussed in [7].

Appendix

Theorem: All points in D are correctly classified by the NN rule using subset C generated by Method 2.

Proof:

a) All points from C are classified correctly.

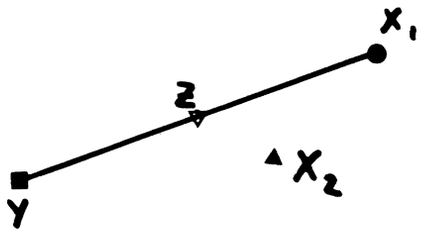

Fig. 7. Illustration for proof in Appendix.

b) Let $x_1 \in D - C_1$ be classified incorrectly. Let $y \in C$ be its nearest neighbor. Assume $x \in$ class 1, so that $y \in$ class 2. Let $z = 0.5*(x_1 + y)$ (Fig. 7). Since $x_1 \in C$ there must be a point $x_2 \in D$ closer to z than x_1 (otherwise x would belong to C by definition). $x_2 \in C$ is impossible since y is nearest to x_1 of all points of C and dist $(x_2,x_1) <$ dist (y,x_1).

Now either

1) $x_2 \in$ class 2. Since $x_2 \in C$ this means that there is another point ($x_3 \in D$) inside $S(x_1,x_2) \subset (x_1,y)$; otherwise $x_2 \in C$ which is a contradiction.

($S(x_1,x_2)$ is the bigger sphere centered at $0.5*(x_1 + x_2)$ with radius $0.5*$ dist (x_1,x_2)). This either leads us back to the beginning of argument 1 with x_2 replaced by x_3, or to case 2. In both situations we arrive at a contradiction due to the finite size of D.

2) $x_2 \in C_1$, and we are in the same situation as at the beginning of the proof, with x_1 replaced by x_2, but dist $(x_2,y) <$ dist (x_1,y). By induction and the finiteness of D it leads to a contradiction.

This completes the proof.

References

[1] P. E. Hart, "The condensed nearest neighbor," *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 515–516, May 1968.
[2] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. New York: John Wiley & Sons, 1973.
[3] R. H. Hammond, W. B. Rogers, and B. Hauck, Jr., *Introduction to FORTRAN IV*. New York: McGraw-Hill, 1976.
[4] G. L. Ritter *et al.*, "An algorithm for a selective nearest neighbor decision rule," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 665–669, Nov. 1975.
[5] G. W. Gates, "The reduced nearest neighbor decision rule," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 431–433, May 1972.
[6] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-2, pp. 408–421, July 1972.
[7] I. Tomek, "An experiment with the edited nearest neighbor rule," *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-6, pp. 448–452, June 1976.

Polynomial Interpolation Errors for Band-Limited Random Signals

ROBERT B. KERR

***Abstract*—One method of digitally simulating a differential system in real time involves approximating the input signals by polynomial interpolations between their sample values. In this note, the statistical error in various polynomial interpolations of band-limited random signals sampled at the Nyquist rate is investigated. The time average of the mean-squared error is calculated for zero-, first-, and second-order holds, with best results obtained for the first-order hold (linear interpolation).**

Introduction

Some approximation is required if one is to simulate the behavior of a differential system, driven by continuous-time analog forcing functions, with a digital computer. One of the

Manuscript received April 7, 1976; revised June 9, 1976.
The author is with the Department of Electrical Engineering, Duke University, Durham, NC 27706.